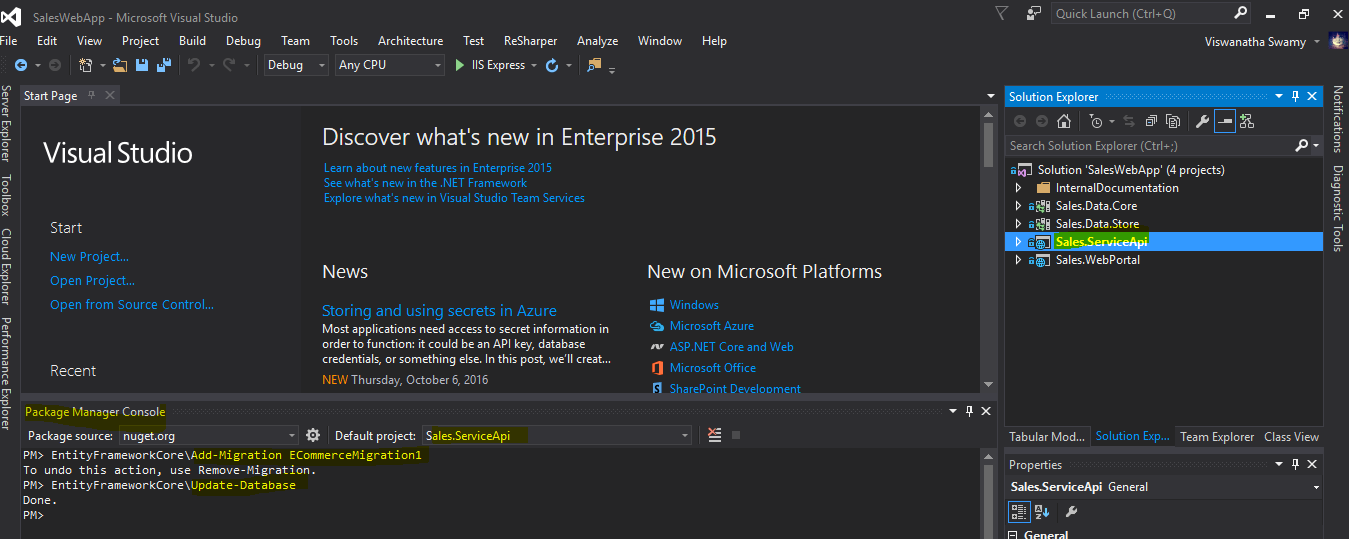
**Step 1:**

**Open the solution; Update the appsettings.json inside Sales.ServiceApi. In our case it is SQL Azure.**

**Step 2:**

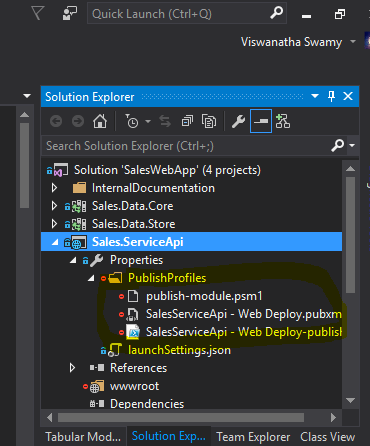
**Set Sales.ServiceApi as Start-up project. Select Sales.ServiceApi in Package Manager Console and execute the below mentioned commands**

* **EntityFrameworkCore\Add-Migration ECommerceMigration1**
* **EntityFrameworkCore\Update-Database**



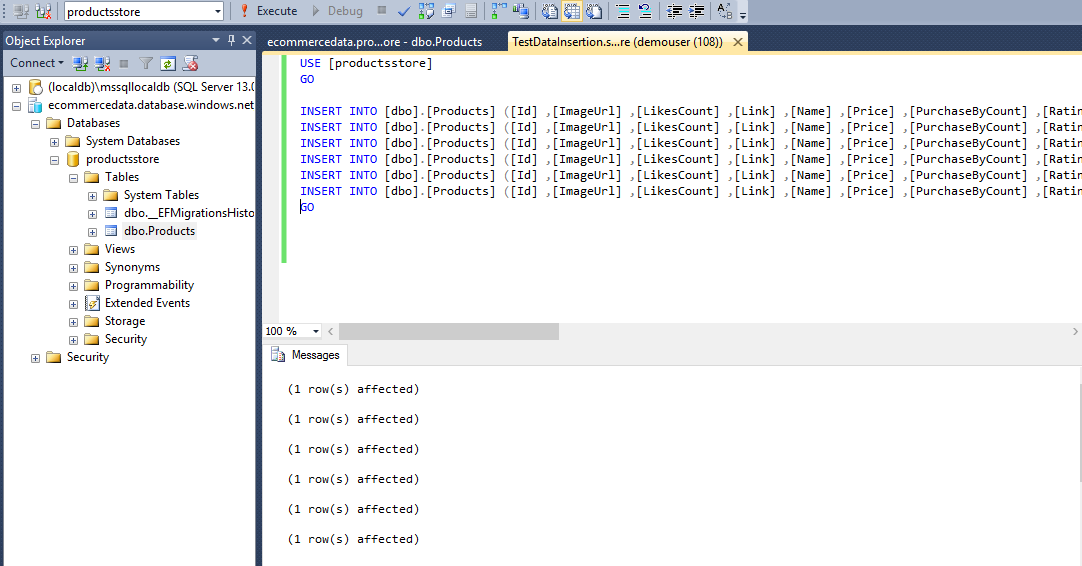
**Step 3:**

**Publish the Web API to Azure Cloud (**<http://salesserviceapi.azurewebsites.net/api/Products>**)**



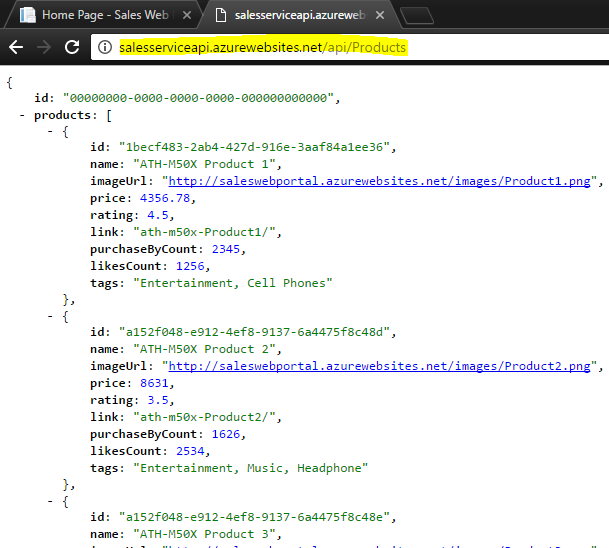
**Step 4:**

**Login into SQL Azure using SQL Management Studio. Execute TestDataInsertion.sql to create test data.**



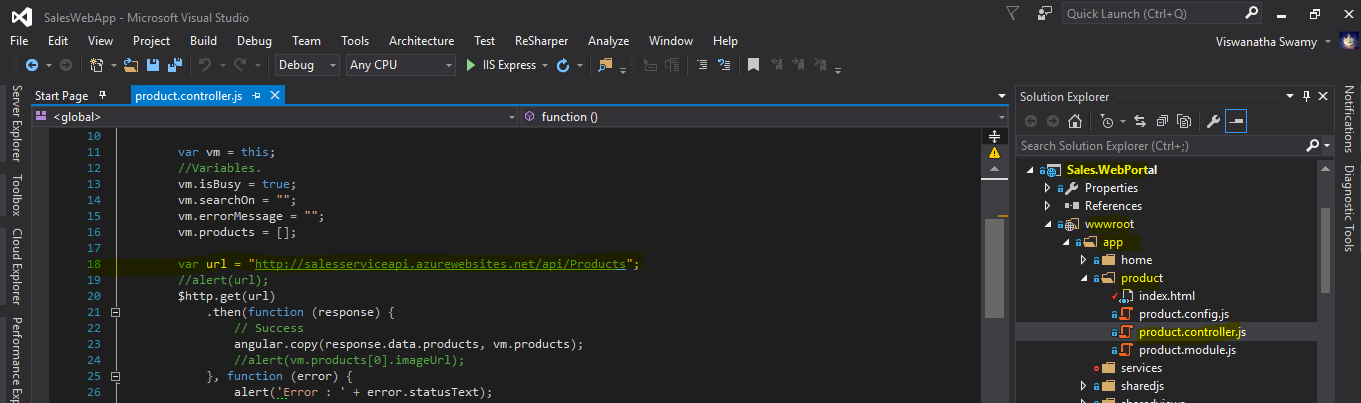
**Step 5:**

**Browse (**<http://salesserviceapi.azurewebsites.net/api/Products>**) to view the data.**



**Step 6:**

**Set Sales.WebPortal as Start-up project. Update the URL on line 18 inside product.controller.js [YourPath\SalesWebApp\Sales.WebPortal\wwwroot\app\product\product.controller.js]**



**[Minimum Requirement]**

— Use of Web API to fetch product detail. products[]

— [ name, image, price, rating, tags[], link, purchased\_by ].

— Visually interactive responsive design listing all the products.

— Archive and submit source code, screenshots, deployment instructions.

**[Plus Point]**

— A feature to **Search** (via **tags**).

— Add autocomplete feature to **Search** product tags conveniently.

— Display **total** products and integrate product details (rating, price, image, tags, etc)

— **Link** url\_address to respective URL. (Should open in new window | target="\_blank")

— Feel free to use your **favourite UI/UX frameworks** and tools in design and development.

**[Extra Work]**

— Awesome if you could manage to deploy/host this webapp at any of your favourite cloud portal and share Demo URL. (i.e. AWS, Heroku, OpenShift, etc)

— Implement **Sort** (by Rating, Price or Like ) feature.

— Custom elegant design, fonts and icons to make web app more user-friendly.

— You may add portfolio activity comprising awesome work you have done on web application(s).

— Use your imagination and add features which would make things easier for end users. Submit link to github or existing hosted apps will help. Please don’t forget to add deployment instructions or READ.ME

**[Guide]**

— Products API:

— Ideal Frontend Stack:

* Responsive Design: HTML5, CSS3 (LESS/SASS, Bootstrap/Foundation)
* Any of your favourite JS Libraries / Framework: jQuery, Angular, Backbone or React.
* Optional: Typography, Subtle pattern, Custom Icons.
* Use Modular / Component Design approach

— PS: Application can be built with use of client-side scripting and need not require any backend.

— IMPORTANT: Archive source, screenshots and documentation in ZIP file and upload it.

— For icons, color and subtle texture:

http://www.flatuicolorpicker.com/orange

http://glyphsearch.com/

http://subtlepatterns.com/thumbnail-view/

http://codebeautify.org/jsonviewer

http://www.flaticon.com/categories

**[Wireframe MVP / Reference Structure]**

Mockup image with instructions

**App Page I** — Home page comprising list of repository and relevant details.

Web API URL: <http://salesserviceapi.azurewebsites.net/api/Products>

Web Application URL: <http://saleswebportal.azurewebsites.net/>

Source Code: <https://github.com/vishipayyallore/ASP-Dotnet-Core-Samples/tree/master/SalesWebApp>

Read Me Path: <https://github.com/vishipayyallore/ASP-Dotnet-Core-Samples/tree/master/SalesWebApp/InternalDocumentation>